

REMARKS

Claims 1-8

Claims 1-8 were rejected under 35 U.S.C. § 102(b) as being anticipated by Sproat et al., "A Stochastic Finite-State Word-Segmentation Algorithm for Chinese," *Computational Linguistics*, vol. 22:3, September 1996 (hereinafter Sproat).

Sproat discloses a system for segmenting text using a finite state machine. The finite state machine defines possible sequences of characters that can form segments and assigns a cost to each segment. During segmentation, competing segmentations for the same sequence of characters are evaluated based on their costs. The lowest cost segmentation for the entire sequence is adopted as the segmentation for the sequence of characters. Sproat also describes a prior art segmentation called the greedy algorithm in which a sequence of characters is segmented from left to right by finding the largest string of characters that form a word beginning with the first character. After the longest word has been found for the first character, the method jumps to the end of that word in the sequence of characters and looks for the longest word that begins with the character after the previously found word. This continues until the end of the sequence of characters is reached.

Independent claim 1 of the present application provides a method for segmenting an input sequence of characters. The method includes identifying possible segments in the sequence of characters with at least two of the possible segments overlapping each other. An alternative sequence of characters is identified for at least one of the possible segments, where the alternative sequence of characters forms an alternative segment that fills the same space as one of the possible segments. Multiple syntactic analyses are then performed using the possible segments and the alternative segments. These analyses result in a full

syntactic parse that utilizes and thereby results in a segmentation of the input sequence of characters.

As amended, claim 1 is patentable over Sproat because Sprout does not identify an alternative sequence of characters for at least one possible segment where the alternative sequence of characters fills the same space as the possible segment. Further, Sproat does not perform multiple syntactic analyses using the possible segments and the alternate segments.

As discussed in the present application on page 16, an alternative segment fills the same space as a possible segment if it is designated as starting and ending at the same character positions as the possible segment. In other words, the alternative sequence is deemed to span the same character positions in the sequence of characters.

Sproat never mentions any system that can identify an alternative sequence of characters for a possible segment in an input sequence of characters where the alternative sequence fills the same space as the possible segment. Under the Sproat systems, the only sequences of characters that can be found are those that are present in the input sequence. Thus, if there is a sequence of characters ABCDE, Sproat will identify segmentations such as "ABCD" "E" or "ABC" "DE". It will not identify an alternative sequence for the segment "ABC" that will fill the same space as "ABC". For example, Sproat will not identify an alternative sequence of "ABF" for the "ABC" segment.

Sproat will also not identify an alternative sequence of "AB" for the segment "ABC" where the "AB" sequence fills the same space as the "ABC" sequence. This can be seen from the fact that if Sproat identified "AB" as a segment, it would have to group character "C" with the next segment. This would mean that "AB" would not fill the same space as "ABC" because it would not extend to the character position of character "C".

Under the invention of claim 1, on the other hand, alternative sequences for a segment are found which are deemed to fill the same space. For example, for a segment ABC, the present invention may find an alternative sequence of ABX or ADXJ that is deemed to fill the same space as one of the segments. Thus, no matter how many characters are in the alternative sequence, it is deemed to start and end at the same position as the possible segment. Sproat does not show or suggest forming such alternative sequences of characters.

In addition, Sproat does not perform syntactic analyses using a possible segment and an alternative segment to produce a segmentation of an input sequence of characters. In the Office Action, it asserted that Sproat showed such syntactic analyses on pages 382-384 and 393-394. However, in the cited sections, Sproat makes no mention of performing syntactic analyses or of forming a segmentation by performing syntactic analyses. Note that under Sproat, the segmentation is performed directly using the finite state machine and selecting the string that produces the lowest cost. No syntactic analysis is performed.

Since Sproat does not show identifying an alternative sequence of characters and does not perform multiple syntactic analyses to produce a segmentation of an input sequence of characters, it does not show or suggest the invention of claim 1 or claims 2-8 which depend therefrom.

Claims 9-14

Claims 9-14 were rejected under 35 U.S.C. § 102(b) as being anticipated by International Patent WO 98/08169 to Carus et al. (hereinafter Carus).

Carus provides a segmentation system that uses three layers of segmentation. In the first layer, heuristic rules are applied to the input sequence to identify possible segments based on things such as numbers, punctuations and change in script. At the second layer, a statistical analysis model is used that

searches for character strings from two to four characters in length that are known to be linked together or that are known to always be broken apart. The links or breaks between the characters are inserted at this level. At the third level, the portions of the character string that have not been broken or linked together are compared to a dictionary to determine if they appear in the dictionary. Segments are then formed based on the portions of the character string that match the dictionary.

Independent claim 9 provides a system for identifying syntax in a string of characters from a non-segmented language. The system includes a word breaker that generates a collection of words from the string of characters. The collection of words includes at least two words that are derived in part from the same character in the string of characters. The word breaker utilizes a lexical record set that is used to derive words for the collection of words by taking the words directly from the string of characters. The word breaker also uses a variants constructor to derive word variants that are not found in the string of characters. Each variant is added to the collection of words and has a different sequence of characters than the sequence of characters associated with the word in the string of characters from which it is derived. A syntax parser performs a syntactic analysis using the collection of words produced by the word breaker to produce a syntax parse. The syntax parse indicates the syntax of the string of characters.

Carus does not show or suggest the invention of claim 9 because it does not show or suggest a variants constructor that is used to derive word variants that are not found in the string of characters. Further, Carus does not include a syntax parser that performs a syntactic analysis using the collection of words produced by the word breaker to produce a syntax parse.

In the Office Action, page 36 and FIGS. 5 and 6 were cited as showing a variants constructor. However, it is clear

that the cited sections do not show a variants constructor that is used to derive word variants that are not found in the string of characters. In the cited sections, Carus is simply searching a dictionary for sub-strings of characters found in the input sequence of characters. Carus is not trying to derive word variants that are not found in the string of characters.

Carus also fails to show a syntactic parser that performs a syntactic analysis. In the Office Action, FIG. 2 and page 9, lines 22-36 and page 10, lines 1-9 were cited as showing a syntactic parser. The cited sections do not mention a syntactic parser. Instead, they discuss heuristic rules that are applied to an input sequence to identify links and breaks in the sequence and thereby identify segments in the input sequence. There is no mention of a syntactic analysis, a syntactic parse, or an identification of syntactic units.

Since Carus does not show a variants constructor that is used to derive word variants that are not found in the string of characters or a syntactic parser, it does not show or suggest the invention of claim 9 or claims 10-14 which depend therefrom.

Claim 11

Claim 11 was rejected under 35 U.S.C. § 103(a) as being unpatentable over Carus in view of Sproat.

Claim 11 depends from claim 9 and includes a further limitation where the variants constructor uses inflectional morphology rules that are capable of identifying a word's lemma from its inflectional form in the string of characters.

In the Office Action, it was asserted that Sproat discloses inflectional morphology rules that can identify a word's stem at pages 386 and 398. After reviewing these pages, Applicant has been unable to find any reference to morphology rules that identify word stems. Instead, Sproat simply discusses morphology rules in which plurals of nouns are formed. It does

not attempt to identify a stem or lemma of a plural word in the sequence of characters.

In addition, even if Sproat did show morphology rules for identifying a lemma, it would not be obvious to combine the identification of a words lemma with the system in Carus. In particular, Carus is only focused on segmenting a character string based on the characters in the string. Even if a lemma were identified for characters in the string, Carus has no way of using the lemma to perform further segmentation. As such, those skilled the art would not be motivated to combine morphology rules for identifying a lemma with Carus.

In addition, Sproat also fails to show a variants constructor that is used to drive word variants that are not found in the string of characters. Under Sproat, only the characters present in the input string are identified as possible segments or words. There is no mechanism in Sproat for deriving a word variant that is not found in the string of characters. As such, the combination the Carus and Sproat fails to show a variants constructor as found in claim 9 and thus as found in Claim 11, which depends from claim 9.

Since neither Carus nor Sproat show a variants constructor and since Sproat does not show morphology rules that identify a words lemma and further since there would be no motivation for combining such morphology rules with Carus, the invention of claim 11 is patentable over the combination of Carus and Sproat.

Claims 15-24

Claims 15-24 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Carus in view of Sproat.

Independent claim 15 is directed to a computer-readable medium having instructions for performing steps that include receiving a sequence of characters that represent a phrase for a non-segmented language. A variant for a first group of characters

is identified. The variant contains a different collection of characters than the collection of characters from the first group of characters while being considered to fill the space of the first group of characters. A second group of characters is then identified in the sequence of characters that overlaps the first group of characters. A syntactic analysis is performed using the variant and the second group of characters to produce a syntactic parse. The syntactic parse contains either the variant or the second group of characters.

The invention in claim 15 is not shown or suggested in the combination of Carus and Sproat. In particular, neither reference identifies a variant for a first group of characters in the sequence of characters, where the variant contains a different collection of characters than the collection of characters in the first group of characters while being considered to fill the space of the first group of characters. As noted above, formation of this type of variant is simply not discussed in either Sproat or Carus. Further, neither reference shows performing a syntactic analysis using a variant to produce a syntactic parse that includes either the variant or a second group of characters.

Since neither Carus nor Sproat show a step of identifying a variant for a group of characters, their combination does not show or suggest the invention of claim 15 or claim 16-24, which depend therefrom.

Conclusion

Although the patentability of specific independent claims has been discussed above, Applicants do not agree with the rejection of the dependent claims nor the reasons for those rejections. However, since the independent claims have been amended, the rejections of the dependent claims are no longer consistent with the current form of the independent claims. As such, discussions regarding the patentability of the dependent

claims would be confusing at this point. Nonetheless, all of the dependent claims are patentable for the reasons discussed above for the independent claims.


In light of the above remarks, claims 1-24 are patentable over the cited references. Reconsideration and allowance of the claims is respectfully requested.

The Director is authorized to charge any fee deficiency required by this paper or credit any overpayment to Deposit Account No. 23-1123.

Respectfully submitted,

WESTMAN, CHAMPLIN & KELLY, P.A.

By:



Theodore M. Magee, Reg. No. 39,758
Suite 1600 - International Centre
900 Second Avenue South
Minneapolis, Minnesota 55402-3319
Phone: (612) 334-3222 Fax: (612) 334-3312

TMM:sew